Deploying Approaches for Pattern Refinement in Text Mining

Sheng-Tang Wu Yuefeng Li Yue Xu
School of Software Engineering and Data Communications
Queensland University of Technology, QLD 4001 Australia
{*s.wu, y2.li, yue.xu*}*@qut.edu.au*

Abstract

Text mining is the technique that helps users find useful information from a large amount of digital text documents on the Web or databases. Instead of the keyword-based approach which is typically used in this field, the pattern-based model containing frequent sequential patterns is employed to perform the same concept of tasks. However, how to effectively use these discovered patterns is still a big challenge. In this study, we propose two approaches based on the use of pattern deploying strategies. The performance of the pattern deploying algorithms for text mining is investigated on the Reuters dataset RCV1 and the results show that the effectiveness is improved by using our proposed pattern refinement approaches.

1. Introduction

Text mining is the technique that helps users find useful information from a large amount of digital text documents on the Web or databases. It is therefore crucial that a good text mining model should retrieve the information that meets users' needs within a relatively efficient time frame. Traditional Information Retrieval (IR) has the same goal of automatically retrieving relevant documents as many as possible while filtering out non-relevant ones at the same time [5]. However, IR-based systems cannot meet users' needs [11].

Many text mining methods have been developed in order to achieve the goal of retrieving useful information for users [4]. Most of them adopt the keyword-based approach, whereas the others choose the phrase-based technique to construct a text representative for a set of documents. Lewis [9] conducted experiments using phrasal indexing language on a text categorization task, but the results showed that performance for a phrase-based indexing language was lower than that for a word-based one. Although phrases contain less ambiguous and narrower meanings than individual words, the likely reasons for the discouraging performance from the use of phrases are: (1) phrases have inferior statistical properties to words, (2) they have low frequency of occurrence, and (3) there are a large number of redundant and noisy phrases among them [15]. Scott and Matwin [14] also suggested that simple phrase-based representations are not worth to pursue since they found no significant performance improvement on eight different representations based on words, phrases, synonyms and hypernyms. They concluded that combining classifiers with alternative representations can produce better results.

In our previous work [18], the experimental results showed that Pattern Taxonomy Model (PTM) is a feasible way to apply data mining techniques to the text mining area. However, it is obviously not a desired method for conquering the challenge because of its low capability of dealing with the mined patterns. In our opinion, more robust and effective pattern deploying techniques need to be implemented. Therefore, in this paper we propose two novel pattern deploying algorithms to effectively exploit discovered patterns for the text mining problem.

The rest of this paper is structured as follows. Section 2 describes the related works and the terminology used in this study is presented in section 3. Following is the discussion of experimental setting and results evaluation. Finally, Section 5 concludes this study work and discusses some future works.

2. Related Works

In [10], term frequency and inverse document frequency (tfidf) weighting scheme is used for text representation in Rocchio classifiers. In addition to tfidf, the global idf and entropy weighting scheme is proposed by [3] and improves the performance by an average of 30%. Varying weighting schemes for the *bag of words* representation approach are given in [7]. The problem of the *bag of words* approach is how to select a limited number of features among an enormous set of words or terms in order to increase the system's efficiency and avoid the *overfitting* [15]. To reduce

Proceedings of the Sixth International Conference on Data Mining (ICDM'06)
0-7695-2701-9/06 $20.00 © 2006 IEEE

the number of features, many dimensionality reduction approaches have been conducted by the use of feature selection techniques, such as *Information Gain*, *Mutual Information*, *Chi-Square*, *Odds ratio*, and so on. The details of these selection functions are stated in [15].

The choice of a representation depends on what one regards as the meaningful units of text and the meaningful natural language rules for the combination of these units [15]. With respect to the representation of the content of documents, some research works have used phrases rather than individual words. In [2], combination of *unigram* and *2-gram* is chosen for document indexing in text categorization (TC) and evaluated on a variety of feature selection functions (FEF). Sharma et al. [16] propose a phrase-based text representation for web document management using rule-based Natural Language Processing (NLP) and Context Free Grammar (CFG) techniques. In [1], Ahonen et al. apply data mining techniques to text analysis by extracting co-occurring terms as descriptive phrases from document collections.

3. Deploying Method

There are several ways to utilize discovered patterns by using a weighting function to assign a value for each pattern according to its frequency. One strategy has been implemented and evaluated in [18], which proposed a pattern mining method that treated each found sequential pattern as a whole item without breaking them into a set of individual terms. Each mined sequential pattern p in PTM was given a value based on the following weighting function:

$$W(p) = \frac{|\{d_a | d_a \in D^+, p \text{ in } d_a\}|}{|\{d_b | d_b \in D, p \text{ in } d_b\}|} \quad (1)$$

where d_a and d_b denote documents, and D^+ indicates positive documents in D, such that $D^+ \subseteq D$. However, the problem of this method was the low pattern frequency due to the fact that it is difficult to match patterns in documents especially when the length of the pattern is long. Therefore, a proper pattern deploying method to overcome the low pattern frequency problem is needed. We propose two methods to deploy discovered patterns in this study. The first one is pattern deploying method (PDM) and the second one is pattern deploying with relevance function (PDR).

3.1 Pattern Deploying Method (PDM)

As mentioned above, a method is needed to deploy sequential patterns into a feature space. The relation among found patterns can be described as "is-a" relation ($\sqsubseteq$) in pattern taxonomies using PTM. Hence, there are likely many overlaps among these patterns [17]. To represent the overlaps among patterns, we deploy the set of patterns for the document d_k on T, the set of terms, to obtain the following vector:

$$\vec{d_k} =< (t_{k_i}, n_{k_i}), (t_{k_2}, n_{k_2}), \ldots, (t_{k_m}, n_{k_m}) > \quad (2)$$

where t_i in pair (t_i, n_i) denotes a single term and n_i is its support in d_k which is the number of patterns that contain t_i.

The detailed deploying process is presented in Algorithm 1. Note that the SPMining algorithm [18] is used in line 3 for generating frequent sequential patterns. The main process of pattern deploying is between line 5 and line 7 inclusively. The normalization in line 8 is normally used to assume the contribution of each document is equal. The result of this algorithm is a set of documents, which can be expressed as follows.

$$\eta = \{\vec{d_1}, \vec{d_2}, \ldots, \vec{d_n}\} \quad (3)$$

We also need to determine the weight for each term in T when we use the discovered knowledge in η. The weighting scheme for a given term t is denoted as the following function.

$$weight(t_i) = \sum_{\vec{d_k} \in \eta} \left(\frac{n_{k_i}}{\sum_{(t,w) \in \vec{d_k}} w}\right) \quad (4)$$

where w is the support of term t indicating the number of patterns that contain t in a document d.

In order to deploy discovered patterns and acquire deployed support of each term in these patterns, we use a pattern composition operation to join two patterns. Let $termset(P) = \{t | (t, f) \in P\}$ be the termset of pattern P, where f denotes term frequency in pattern P. The composition of two patterns P_1 and P_2 can be processed by using the following expression.

$$\begin{aligned} &P_1 \oplus P_2 = \{(t, f_1 + f_2) | (t, f_1) \in P_1, (t, f_2) \in P_2\} \cup \\ &\{(t, f) | t \in (termset(P_1) \cup termset(P_2)) - \\ &(termset(P_1) \cap termset(P_2)), (t, f) \in P_1 \cup P_2\} \end{aligned} \quad (5)$$

For example, $P_a = \langle t_1, t_2 \rangle$ and $P_b = \langle t_2, t_3 \rangle$ are two frequent patterns. The termsets of P_a and P_b are $\{(t_1, 1), (t_2, 1)\}$ and $\{(t_2, 1), (t_3, 1)\}$ respectively. The composition of these two patterns $P_a \oplus P_b$ can be denoted as P' where $termset(P') = \{(t_1, 1), (t_2, 2), (t_3, 1)\}$.

Algorithm 1. PDM(*D*, *min_sup*)
Input: a list of documents, *D*; minimum support, *min_sup*.
Output: a set of documents η.
Method:

1: $\eta \leftarrow \{\varnothing\}$
2: **foreach** document d in D **do begin**
3: SP = SPMining(*PL*, *min_sup*)
4: $\nu \leftarrow \{\varnothing\}$
5: **foreach** pattern P in SP **do begin**
6: $\nu \leftarrow \nu \oplus P$ *// pattern composition*
7: **end for**
8: normalize ν
9: $\eta \leftarrow \eta \cup \nu$
10: **end for**

3.2 Pattern Deploying with Relevance Function (PDR)

PDR is a deploying method with the use of relevance function. This approach utilizes a probabilistic method to estimate the term weight. By using SPMining algorithm [18], we acquire a set of frequent sequential patterns Ω from each document, such that $\Omega = \{p_1, p_2, \ldots, p_n\}$, where $p = \langle (t_1, f_1), (t_2, f_2), \ldots, (t_m, f_m) \rangle$ and f_i is the term frequency of term t_i. Assume that we have a hierarchy of all keywords in T, which consists of a set of clusters Θ. Each cluster in Θ is represented as a term, such that $\Theta \subseteq T$. The support of p in Ω can be described as $support :: \Omega \rightarrow [0, 1]$, such that

$$support(p) = \frac{supp_a(p) \times I(p)}{\sum_{p_i \in \Omega} supp_a(p_i)} \quad (6)$$

where $support(p)$ indicates the importance to the document that contains it and $I(p)$ is a length reinforcement derived by $(len(p))^2$. The greater the support is, the more important the pattern p is. To deploy patterns to a term space Θ, a mapping β to explicitly describe the relationship between pattern and the term space is used and defined as $\beta :: \Omega \rightarrow 2^{\Theta \times [0,1]} - \{\varnothing\}$, such that

$$\beta(p) = \{(t_1, w_1), (t_2, w_2), \ldots, (t_n, w_n)\} \subseteq \Theta \times [0, 1] \quad (7)$$

where w_i is the term weight of t_i. Assume that all terms belonged to a certain pattern have the same weight so that w in the above mapping can be derived by the following function: $w_i = 1/len(p)$. The probability of a term t can be described by using the following function.

$$pr_\beta(t) = \sum_{p \in \Omega, (t,w) \in \beta(p)} support(p) \times w \quad (8)$$

A relevance function for a document d can be defined as follows.

$$relevance(d) = \sum pr_\beta(t)\tau(t, d)$$

where

$$\tau(t, d) = \begin{cases} 1 & \text{if } \ t \in d \\ 0 & \text{otherwise.} \end{cases} \quad (9)$$

The procedure of PDR is given in pseudo codes in Algorithm 2. The output of PDR is the same as the one from PDM. The main difference between PDM and PDR is that in the latter method, the support of pattern is involved in the estimation of term weights (line 2 and 5 in Algorithm 2) but not in the former method.

Algorithm 2. PDR(*SP*, ν)
Input: a set of frequent sequential patterns, *SP*; a set of pairs of term and its weight, ν.
Output: an updated set of pairs of term and its weight, ν.
Method:

1: **foreach** pattern P in SP **do begin**
2: sum_supp += $supp_a(P)$
3: **end for**
4: **foreach** pattern P in SP **do begin**
5: $weight' = supp_a(P)/sum\_supp/len(P)$
6: **foreach** term t in P **do begin**
7: **if** $(t, weight) \in \nu$ **then**
8: $(t, weight) = (t, weight + weight')$
9: **else**
10: $\nu \leftarrow \nu \cup (t, weight')$
11: **end if**
12: **end for**
13: **end for**

4. Experiments and Results

In this study we use the Reuters text collection to evaluate the proposed approaches. Term stemming and stopwords removal techniques are used in the prior stage of text preprocessing. We then apply several common measures for performance evaluation and compare our results with Pr, Rocchio and PTM methods. The details of these methods are discussed in Section 4.3.

4.1 Experimental Dataset

There are several standard benchmark datasets available in text mining area, including Reuters corpora[1], OHSUMED [6], and 20 Newsgroups collection [8]. The most frequently used one is the series of Rueters dataset. The particular version that we chose for this experiment is

[1] http://about.reuters.com/researchandstandards/corpus/

Reuters Corpus Volume 1, also known as RCV1 . The reasons are that the RCV1 is the latest one among these common data collections and it also contains a reasonable number of documents with reference judgment for the benchmark. Although another version of Rueters-21578 is currently the most widely used dataset for text categorization research, it is believed with high possibility to be superseded by RCV1 over the next few years. In this study we chose all 100 TREC topics (topic 101 to topic 200) for our evaluation. Further details regarding the RCV1 can be found in [13].

4.2 Data Preprocessing and Measures

Data preprocessing is applied before the documents can be interpreted by the deploying methods. The fields we chose in a document are "title" and "text". The content in "title" is viewed as a paragraph as the one in "text" which consists of paragraphs. For dimensionality reduction, stopwords removal is applied and Porter algorithm [12] is selected for suffix stripping. Terms with frequency equaling to one are discarded. To evaluate experimental results, we used several standard measures such as the precision of first k returned documents (*top-K*), breakeven point (b/e), averaged precision ($P_{avg.}$), F_β-*measure* with $\beta = 1$ (F_1), and the first value in the interpolated eleven points (11 $pt.$).

4.3 Results and Discussion

For evaluating the proposed algorithm, we apply the pattern deploy method, PDR, to the information filtering task. For each topic, the system extracts the concept and aims to filter out the non-relevant incoming documents according to the user profiles. Concept generating is based on the Rocchio algorithm which is used to build the profile for representing a concept of a topic which consists of a set of relevant and irrelevant documents. The Centroid $\vec{C}$ of a topic can be generated by using the following Rocchio equation.

$$\vec{c} = \alpha \frac{1}{|D^+|} \sum_{\vec{d} \in D^+} \frac{\vec{d}}{\|\vec{d}\|} - \beta \frac{1}{|D^-|} \sum_{\vec{d} \in D^-} \frac{\vec{d}}{\|\vec{d}\|} \qquad (10)$$

where α and β are empirical parameters; D^+ and D^- are the set of relevant documents and the set of irrelevant documents respectively; d denotes a document.

For comparison purpose, we implement another keyword-based algorithm *Pr* [5] in our experiments. With this heuristic, a term t is weighted using the following formula:

$$W(t) = \log\left(\frac{r + 0.5}{R - r + 0.5} \div \frac{n - r + 0.5}{(N - n) - (R - r) + 0.5}\right) \qquad (11)$$

where N and R are the total number of documents and the number of relevant documents in training set respectively; n is the number of documents which contain t, and r is the number of relevant documents which contain t.

In this study, we only focus on the relevant documents in the training dataset in order to implement all methods in a fair way. This is because the PTM used relevant documents only during the training phase. Therefore, the value of α can be a constant and the value of β is given as zero.

Four methods used in our experiments are summarized as follows.

- **Pr**: the keyword-based method which uses Equation 11 for term weighting.
- **PTM**: PTM model where $min\_sup = 0.2$ with pattern pruning.
- **Rocchio**: the keyword-based model using Equation 10 to represent the concept of documents.
- **PDR**: proposed method deploying method with relevance function.

By using PDR, it improves the performance of the precision of the first 20 returned documents by around 10.2% against Rocchio method from 48.9% to 53.9% in Table 1. In addition, PDR use less number of training patterns compared to PTM and Rocchio. The number of training patterns has been reduced 39% compared with Rocchio as shown in Table 2. PTM uses the largest number of training patterns, as compared in Table 1, but it seems not to benefit the performance. The main reason is that PTM treats each individual pattern as an element and analyzes its statistical feature directly from the dataset without any further post-processing, which in turn encounters the low pattern frequency problem.

We compare PDR with the other three methods using the measure of interpolated eleven-point average *precision*/*recall* in Figure 1. The PDR method yielded over 80% of precision in the first point where *recall* equals zero. It also outperforms the other three methods on the other ten *recall* points. Based on these observations, we can summarize that our proposed method PDR not only uses less number of training patterns, but also improves the effectiveness.

5. Conclusions

In this study we propose two pattern refinement methods to deploy the discovered patterns into a feature space which is used to represent the concept of documents. Our methods adopt the mining sequential pattern technique to find semantic patterns from text documents and then deploy these patterns using proposed deploying algorithms. The experimental results show that the use of proposed methods

	Pr	PTM	Rocchio	PDR	+(%)
top-20	0.475	0.486	0.489	**0.539**	10.2
$b/e.$	0.419	0.403	0.434	**0.471**	8.6
$P_{avg.}$	0.427	0.411	0.442	**0.479**	8.5
F_1	0.425	0.417	0.436	**0.457**	4.7
11 $pt.$	0.696	0.769	0.728	**0.803**	10.2

Table 1. Averaged results on all 100 RCV1 topics and improvement from Rocchio to PDR.

Topic	First 50	Last 50	All
Pr	32,760	37,418	70,178
PTM	331,680	226,950	558,630
Rocchio	32,760	37,418	70,178
PDR	23,900	18,661	42,561
	-27%	-50%	-39%

Table 2. Number of terms or patterns found at pattern discovering phase.

outperforms Pr, Rocchio and PTM methods. This study also indicates that pattern refinement is the key to improve effectiveness for pattern-based methods. In the future works, we will extend these methods by utilizing the rich information from the non-relevant documents to improve the system in term of effectiveness. An adaptive algorithm of learning the change of the characteristics of context is also another research direction.

References

[1] H. Ahonen, O. Heinonen, M. Klemettinen, and A. I. Verkamo. Applying data mining techniques for descriptive phrase extraction in digital document collections. In *ADL'98*, pages 2–11, 1998.

[2] M. F. Caropreso, S. Matwin, and F. Sebastiani. Statistical phrases in automated text categorization. Technical report, Instituto di Elaborazione dell'Informazione, Technical Report IEI-B4-07-2000, 2000.

[3] S. T. Dumais. Improving the retrieval of information from external sources. *Behavior Research Methods, Instruments, & Computers*, 23(2):229–236, 1991.

[4] L. Edda and K. Jorg. Text categorization with support vector machines. how to represent texts in input space? *Machine Learning*, 46:423–444, 2002.

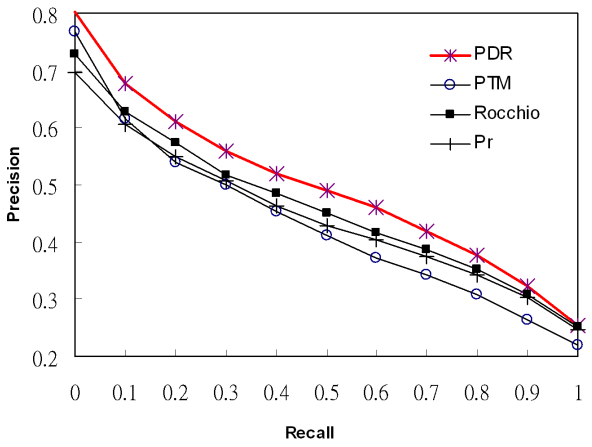

Figure 1. Averaged 11 points plot on all RCV1 topics for all methods.

[5] D. A. Grossman and O. Frieder. *Information retrieval algorithms and heuristics*. Kluwer Academic publishers, 1998.

[6] W. Hersh, C. Buckley, T. Leone, and D. Hickman. Ohsumed: an interactive retrieval evaluation and new large text collection for research. In *SIGIR'94*, pages 192–201, 1994.

[7] T. Joachims. A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. In *ICML'97*, pages 143–151, 1997.

[8] K. Lang. News weeder: Learning to filter netnews. In *ICML'95*, pages 331–339, 1995.

[9] D. D. Lewis. An evaluation of phrasal and clustered representations on a text categorization task. In *SIGIR'92*, pages 37–50, 1992.

[10] X. Li and B. Liu. Learning to classify texts using positive and unlabeled data. In *IJCAI'03*, pages 587–594, 2003.

[11] Y. Li and N. Zhong. Mining ontology for automatically acquiring web user information needs. *IEEE Trans. on Knowledge and Data Engineering*, 18(4):554–568, 2006.

[12] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[13] T. Rose, M. Stevenson, and M. Whitehead. The reuters corpus volume1 - from yesterday's news to today's language resources. In *LREC'02*, pages 29–31, 2002.

[14] S. Scott and S. Matwin. Feature engineering for text classification. In *ICML'99*, pages 379–388, 1999.

[15] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

[16] R. Sharma and S. Raman. Phrase-based text representation for managing the web document. In *ITCC'03*, pages 165–169, 2003.

[17] S.-T. Wu, Y. Li, and Y. Xu. An effective deploying algorithm for using pattern-taxonomy. In *iiWAS'05*, pages 1013–1022, 2005.

[18] S.-T. Wu, Y. Li, Y. Xu, B. Pham, and P. Chen. Automatic pattern-taxonomy extraction for web mining. In *WI'04*, pages 242–248, 2004.